EXPERIMENTS IN SENSOR-BASED COLLISION AVOIDANCE

Homayoun Seraji - Robert Steele - Robert Ivlev
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

Abstract

A control strategy for sensor-based collision avoidance and the supporting experimental results are reported in this paper. The real-time arm control software continuously monitors the object distance measured by the proximity sensors. When this distance is less than a preset threshold, the collision avoidance control action is initiated to inhibit motion towards the object and thus prevent collision. This is accomplished by employing an outer feedback loop to perturb the end-effecter reference motion trajectory in real-time based on the sensory data. The perturbation is generated by a proportional-plus-integral (PI) collision *avoidance controller* acting on the difference between the sensed distance and the user-specified threshold. The proposed approach to collision avoidance is analogous to controlling the *virtual* force applied to the end-effecter by a hypothetical spring-plus-damper attached to the object's surface. This approach is computationally very fast, requires minimal modification to the existing manipulator positioning system, and provides the manipulator with an on-line collision avoidance capability to react autonomously' and intelligently. Experimental results are reported to demonstrate end-effecter collision avoidance with an approaching target and while reaching inside a truss opening,

1 Introduction

Avoidance of collision between a manipulator and objects in its workspace is a basic necessity in any robotic operation. In recent years, ther e has been considerable interest in *sensor-based* collision avoidance in which the data from proximity sensors are utilized *on-line in* the real-time manipulator control system in order t o prevent collision. This approach stands in contrast to *model-based* methods that are used for *off-line* path-planning and are computationally intensive. Boddy and Taylor[1] adopt a reactive approach to sensor-based collision avoidance for redundant manipulators. Volpe and Balaram[2] discuss' a method of

generating a repulsive velocity proportional to the object distance. Cheung and Lumelsky[3] address real-time collision avoidance for whole-sensitive arms. Feddema and Novak[4] describe a collision avoidance system using distributed capacitance-based sensing. Wikman and Newman[5] develop a reflex control approach for oil-line collision avoidance.

This papers describes a novel approach to sensor-based collision avoidance. This approach is based on the concept of controlling the *virtual* forces that are applied to the end-effector when it is at close proximity to an object. An outer collision avoidance loop is closed around the inner manipulator positioning system. The outer loop modifies the Cartesian position setpoints for the inner loop based on the sensed dista.rice between the end-effecter and the object. This enables the manipulator to react to the proximity sensor data by automatically perturbing the commanded arm motion to prevent collision between the end-effecter and the object. This approach can easily be implemented on position-controlled manipulators with minimal modification to their existing control systems,

The paper is structured as follows. The collision avoidance strategy is described in Section 2. Section 3 discusses the hardware and software aspects of the collision detection system used. Experimental results demonstrating the reflexive behavior of the end-effecter to an approaching target and during entry into a constricted opening are presented in Section 4. Section 5 draws the conclusions from this work and addresses areas of current research.

2 Collision Avoidance Strategy

In this section, we describe an end-effector collision avoidance strategy for a position-controlled robot manipulator. Proximity sensors are mounted on the cmd-effecter to detect objects in the Cartesian $x, y,$ and z directions in the end-effecter frame $\{E\}$ attached to the endpoint. Each sensor produces an output proportional to the end-effector distance to the sensed object in a certain direction, and is calibrated so that the sensor reading is the measured object distance.

During free-space motion, the end-effecter Cartesian position coordinates X = $[x, y, z]^T$ track the reference motion trajectories $X_r = [x_r, y_r, z_r]^T$ specified by the user in the fixed world frame {W). The proximity sensors mounted on the end-effector continuously measure the distances $D_m = [d_{mx}, d_{my}, d_{mz}]^T$ between the end-effecter and the workspace objects in the three Cartesian directions x, y, and z of the moving end-effecter frame {E}, The user also specifies the desired stand-off distances $D_r = [d_{rx}, d_{ry}, d_{rz}]^T$ along the three Cartesian axes. Consider, for instance, the end-effecter behavior along the x-axis of the frame {E}. When the end-effecter is further away from the object than the preset distance d_{rx}, i.e., $d_{mx} > d_{rx}$, the motion trajectory is not perturbed and the reference trajectory x_r is executed. On the other hand, when the end-effecter is closer to the object than the threshold, i.e., $d_{mx} \leq d_{rx}$, the reference trajectory x_r is perturbed so as to maintain the stand-off distance d_{rx} to the object, This perturbation is caused by employing the proximity sensor data in the outer feedback loop to perturb the reference motion trajectory x_r in real-time to ensure collision avoidance.

This is accomplished by commanding the end-effecter to deviate by the amount Δx from its nominal trajectory x_r and to track the modified commanded trajectory $x_c = x_r$ — Ax, where Δx is generated by the collision avoidance controller.

The block diagram of the sensor-based end-effecter collision avoidance strategy is shown in Figure 1. The proximity sensor output is modeled as $d_{mx} = x_o - x$ for $x_o > x$, where x_o denotes the object location in the end-effecter frame x-axis and x is the instantaneous value of the end-effecter x-coordinate. When the measured distance d_{mx} is less than or equal to the stand-off distance d_{rx}, the difference $e = d_{rx} - d_{mx}$ is fed to the proportional-plus-integral (PI) collision avoidance *controller* $k_p + k_i \int dt$ to produce the position perturbation Δx as

$$\Delta x = k_p e + k_i \int e dt \tag{1}$$

where k_p and k_i are the constant user-specified proportional and integral gains. The difference e is treated as the *error* and is forced to zero by the collision avoidance controller. Provided that the controller gains are chosen such that the closed-loop system is stable, in the steady-state the integral error $\int e dt$ reaches a constant value. Hence, in the steady-state, e = O and $d_{mx} = d_{rx}$; i.e., the end-effecter maintains the distance d_{rx} to the object. It is important to note that the stand-off distance to the object d_{rx} is preserved in the steady-state despite changes in the object location x_o or the reference position x_r. We conclude that the position perturbation Ax generated by the collision avoidance controller is given by

$$\Delta x = \begin{cases} 0 & \text{for } d_{mx} > d_{rx} \\ k_p[d_{rx} - d_{mx}] + k_i \int [d_{rx} - d_{mx}] dt & \text{for } d_{mx} \leq d_{rx} \end{cases} \tag{2}$$

The above argument can be repeated for collision avoidance in the y and z directions of the end-effecter frame.

Conceptually, we can consider a *hypothetical* spring-plus-damper attached to the object's surface as illustrated in Figure 2a, where k_i is the spring stiffness, k_p is the damping factor, d_{rx} is the natural length, and d_{mx} is the compressed length. Then the *virtual* force applied to the end-effecter is given by

$$F_v = [k_i + k_p \frac{d}{dt}][d_{rx} - d_{mx}] \tag{3}$$

The role of the collision avoidance controller is to cause the virtual force F_v to track the zero force setpoint $F_r = O$ by perturbing the reference trajectory x,. As in conventional position-based explicit force control systems [e.g.,6], the position perturbation Ax is generated by the integral controller $\int dt$, as shown in Figure 2b. '1'bus, the collision avoidance control law is

$$\Delta x = \int [F_v - F_r] dt = [k_p + k_i \int dt][d_{rx} - d_{mx}] \tag{4}$$

which is the proportional-plus-integral controller discussed before. An alternative interpretation is to consider a hypothetical spring with the stiffness coefficient k attached to the object. Then the virtual force applied to the end-effecter by the spring is $F'_v = k[d_{rx} - d_{mx}]$.

To achieve the force setpoint F_r = O, we employ the proportional-plus-integral controller $\frac{1}{k}[k_p + k_i \int dt]$ acting on the force error F_v -- F_r. We conclude that, using the proposed approach, the collision avoidance problem is reformulated and solved as a classical force control problem. This approach makes the vast literature on force control directly applicable to collision avoidance.

Finally, it must be noted that the proximity sensor measurements $D_m = [d_{mx}, d_{my}, d_{mz}]^T$ are in the moving end-effecter frame $\{E\}$ attached to the endpoint, while the reference motion trajectories $X_r = [x_r, y_r, z_r]^T$ are specified in the stationary world frame {W} fixed in the workspace. Therefore, the position perturbations $\Delta X = [\Delta x, \Delta y, \Delta z]^T$ generated by the collision avoidance controllers must be transformed from {l?} to {W} to modify X_r. This transformation is given by

$$^{w}\Delta X = {}^{w}_{e} R.^{e}\Delta X \tag{5}$$

which only involves the 3x3 rotation matrix ${}^{w}_{e}R$ from $\{E\}$ to {W} that is available from the forward kinematic calculations, see[6]. Figure 3 shows the implementation block diagram of the collision avoidance strategy, including the coordinate transformation.

3 Collision Detection System

The collision detection system used in the laboratory is manufactured by the Merritt Systems Incorporated, and is called the Sensor Skin. It is designed to be installed on any of the Robotics Research Corporation (RRC) robot arms. The Sensor Skin system is composed of a collection of circuit boards of which there are two kinds, trapezoidal boards and octagonal boards. There are four basic regions on the RRC arm where the Sensor Skin boards are placed: upper-arm, elbow, forearm, and end-effecter. These boards are shaped in such a way that they can be placed in a circular fashion around the four basic. regions of the RRC arm. Figure 4 shows the mounting of the boards on the RRC model K-1207 arm. Six trapezoidal boards are placed on the upper-arm, two octagonal boards on the elbow (one on each side), five trapezoidal boards on the forearm, and four octagonal boards on the end-effecter (placed at 90° spacing with respect to each other). This mounting scheme allows full coverage of the arm and creates a sensor envelope that protects the arm from intrusion of objects that are in close proximity.

Each board, whether it be a trapezoidal board or an octagonal board, has on it numerous infrared emitters and detectors. 'There are five emitters and five detectors on each octagonal board and three emitters and eight detectors on each trapezoidal board. Therefore, the arm surface is populated with 63 infrared emitters and 118 infrared detectors. The collection of detectors and emitters on an individual Sensor Skin board all report to a single microprocessor chip located on that same board. This collection of sensors, detectors, and microprocessor is referred to hereafter as a Sensor Cell, The Sensor Cell is arranged on the Sensor Skin board in a way that allows for hemispheric coverage of the boar d's surface.

3.1 Sensor Cell Development

A large number of sensors are required in order to instrument an entire robot arm with proximity sensors. Directly interfacing a host computer to such a large array can be quite complex. Merritt Systems solved the problem by utilizing distributed signal processing. The individual detectors on each board share common analog circuitry and each detector reports to a local microprocessor. The microprocessor is the heart of each Sensor Cell. These microprocessors, in turn, communicate with other Sensor Cells via a digital multidrop line or party line. This technique of distributed processing and localized analog circuitry greatly reduces the number of wires needed in order to instrument a robot arm.

The sensing technique selected for the Sensor Cells is based on measuring the amplitude of the reflected infrared light from an object's surface. This technique is selected because of its low cost and simplicity. Figure 5 shows the layout of infrared emitters and detectors on the trapezoidal and octagonal boards. The emitters and detectors are designed to operate at a wavelength of 880nm. The infrared emitters are modulated at a frequency of 31250 Hz. This frequency is chosen to minimize the effect of ambient interference from outside sources such as florescent lights. The infrared detectors are multiplexed to minimize *cross* interference between sensors. When a Sensor Cell is scanning for obstacles, the microprocessor first activates all the emitters on the board, and then each detector is polled sequentially. The emitters are turned off when the scanning process for that board is completed.

3.2 Computer Interface

The Sensor Skin communicates with the user's host computer via two serial RS-485 lines. One line is configured as a multidrop line, and the other is configured as a serial line.

The multidrop line is routed between the user's host computer and the Sensor Skin. This line allows the host computer to command all of the Sensor Cells simultaneously, and receive proximity detection information from reporting Sensor Cells. When an object is detected, only the affected cells will report, all others will remain silent. A threshold value is set on each detector so that a cell will only report when an obstacle is detected. This technique helps to reduce the amount of traffic on the multidrop line, The Sensor Skin can support a two dimensional array of 32 by 32 Sensor Cells on the multidrop line.

The serial channel is a sequential bus that connects the user's host computer to all of the Sensor Cells. This bus is unidirectional, data is sent to one board and then, in turn. it passes that information on to the next, and so on. This allows each Sensor Cell to have a unique position or address in the sensor chain. Upon power up of the Sensor Skin, each Sensor Cell assigns itself an address based on its position in the chain, as shown in Figure 6. This self-addressing scheme allows flexibility in the configuration of the system. Boards can be added or removed as needed in order to accommodate different user configurations,

3.3 Communication Protocols

As described earlier, the Sensor Skin is designed to communicate using the RS-485 protocol. The two serial lines from the Sensor Skin are connected to a VMIC 6015 serial interface card installed in the VME chassis. In our implementation, data is received on the multi-drop serial line, while commands are transmitted on the serial-port line and are echoed back on the same serial line at 4800 baud.

The VMIC 6015 serial card is configured to store each character received from the Sensor Skin and generate an interrupt on the VME backplane. Upon the receipt of this interrupt, the character received is stored into a buffer in local memory and a software is signalled through a semaphore indicating this event. Upon receiving this signal, the software processes the data while the serial card is awaiting new data from the Sensor Skin. The interrupt mechanism allows the system to process the received data in real-time without the need for any polling mechanism or relying upon any synchronization scheme. Due to the low bandwidth of the Sensor Skin, the impact of this interrupt scheme on system performance is minimal.

The data received from the sensor skin is processed by a finite state machine. When a complete data set has been received, the set is moved into the arm control system shared memory, Upon receipt of a request for status information from the user interface, this information packaged with the other arm control status are transmitted to the user over the existing protocol. Upon receipt of the sensor information, the user interface software displays the information to the operator in a graphical format.

The software to communicate with the Sensor Skin consists of the following units:

- Driver to provide the interface with the VMIC 6015 board.
- Unit to read and parse the input data.
- Unit for accepting the parsed data sets and computing the approximate distance of object detected.
- Unit for issuing commands and controlling the sequential scanning operation. This unit is also responsible for placing the Sensor Skin data into shared memory, where it is visible by the rest of the arm control system.

By using the first three software units described above, the last unit controls the actual timing of reading the Sensor Skin. This software unit has the following form:

```
Initialize Skin
Initialize Scan
loop
    Wait for Completed Scan
    Buffer Data
    Synthesize Distance Information
```

```
        Store Results into Shared Memory
        Initiate Scan
    end loop
```

Each scan of the Sensor Skin in the configurate ion used requires approximately 0.3 - 1.0 seconds to complete. This value varies with the number of sensors that are reporting above their preset thresholds.

3.4 Calibration Method

To calibrate each sensor, a test fixture is usedat a know location. The target used for the calibration is a sheet of white paper measuring 12 by 20 inches. 'l'his target was chosen at the manufacturer's suggestion as being the optimal target for the Sensor Skin. The robot is used to move each sensor mounted on the end-effecter perpendicular to the target while recording the sensor reported values and the distances from the target with the Stethoscope software. This information is then used to construct a table of values for determining the distance based upon the sensor reading. Because of the different responses obtained from different sensors, a unique table is required for each sensor on each board used in the experiment.

Figure 7 is the experimental plot obtained for calibration of the outward looking sensor on the sensor board number 7. The distance of the sensor from the target is plotted on the horizontal axis and the sensor measurement value is plotted along the vertical axis. Of the two curves on the plot, one is the result of moving towards the target and the other is the result of moving away from the target. The calibration factors used, represented by the + symbols, are derived by visually approximating the mean distance for a specified measurement value.

Figure 8 plots the error for the given sensor. For this plot, the calibration values obtained are used to run the system using the same trajectory and target as in the calibration procedure, The two curves in Figure 8 correspond to motion towards and away from the target, and the + symbols denote the ideal sensor readings. For a perfectly calibrated sensor, the slope of the curve should be equal to 1. Within the working range of the sensor (i.e., 0.2 to 0.4 meters), this is approximately true.

It must be noted that since the sensors operate based on the intensity of the reflected infrared light, the range of operation of each proximity sensor is heavily dependent on the infrared reflectivity of the target surface. It is found experimentally that the range of operation for a white surface is 20 to 50 cm, whereas for a black surface the range is 2'0 to 25 cm. These represent the two extreme cases and in-between values are found for colored surfaces.

3.5 Infrared Range Sensors

Mounted inside the end-effecter housing are two infrared range sensors manufactured by IDEC. These sensors measure directly the object distance in the z-direction of the end-effector frame by using the triangulation method. This method is superior to the intensity

method used for the Sensor Skin, and provides accurate distance measurements irrespective of the color, material, and shape of the object. However, the main disadvantage of this sensor over the Sensor Skin is its bulkiness, since it measures approximately 5x5x2 cm. These sensors are mounted on opposite sides of the gripper mechanism. The range of operation of these sensors is 20 to 50 cm. In the integration of the sensor data, the sensor reporting the smaller distance from the object is used for collision avoidance. The range sensor data is read and deposited into the shared memory in real-time, and the arm control system accesses this data during run-time.

4 Experimental Results

In this section, we describe two experiments in sensor-based collision avoidance conducted at the JPL Remote Surface Inspection (RSI) Laboratory. These experiments use only the hexagonal sensor boards which are mounted on the end- effector. Of these four sensor units, only the single outward looking sensor of each board is utilized for collision avoidance, These four sensors measure object distances in the $\pm x$ and $\pm y$ directions of the end-effecter frame $\{E\}$ attached to the endpoint. In addition, the range sensor measures the object distance in the $+z$ direction of the end-effecter frame. As a result, object to end-effector distances are continuously measured in five Cartesian directions.

Two experiments on collision avoidance are now described.

4.1 Avoiding an Approaching Target

The purpose of this experiment is to investigate the reflexive behavior of the end-effecter in response to an approaching target.

in this experiment, the end-effecter is stationary and the inner position control loop is servoing to the current position. In the outer collision avoidance loop, the proportional-plus-integral (PI) controller $[k.k_p + k.k_i \int dt]$ is used, where k_p and k_i are chosen as $0{,}05 \times 10^{-3}$ and 1.5x10-3, respectively, and the units used are meters and Newtons. These numerical values were used previously for position-based explicit force control of the RRC arm in a recent experimental study on contact control schemes[7]. A suitable value for the multiplier k is found empirically to be 2×10^3. Therefore, the end-effector collision avoidance problem is reformulated as controlling the virtual force $F_v = -k[d_{rx} - d_{mx}]$ applied to the end-effector by a hypothetical spring with the stiffness coefficient $k = 2000\, Nt/m$.

A white rectangular target is now moved in two steps towards the end-effecter from the side so that it can be detected by the sensor board mounted on the $+x$ surface of the end-effecter. The stand-off distance is set to $d_{rx} = 0.30$ meters. The experimental results are recorded in Figures 9a-9c. Figure 9a illustrates the reflexive behavior of the end-effecter to avoid collision with the target, It is seen that the end-effecter moves initially by 6 cm and subsequently by another 6 cm to avoid colliding with the target and maintain the preset

stand-off distance. Figure 9b shows the variation of the virtual force F_v with the end-effecter position. It is seen that the force is zero when the target is further away from the end-effector than the threshold $d_{rx} = 0.30$ meter, and increases linearly with x otherwise. The PI controller brings the virtual force to zero, as shown in Figure 9c. Figures 9a-9c clearly demonstrate that the end-effecter reacts autonomously and intelligently to ensure collision avoidance.

The experiment is now repeated with the target approaching in the $-x, \pm y$ and $+z$ directions of the end-effector frame. A sample set of the experimental results obtained is shown in Figures 10a-10b. These plots demonstrate the sensor-based collision avoidance capability of the end-effecter in the $+y$ and $+z$ directions.

Finally, the end-effecter is operated under compliance control[7], which is an implicit force control scheme. In this case, the end-effecter behaves like a spring which restores it to the nominal position when the target is removed. However, since there is no explicit force setpoint in the compliance control formulation, this scheme is not found suitable for collision avoidance.

4.2 Reaching Inside an Opening

The goal of this experiment is to utilize the collision avoidance capability to perform an inspection task that requires reaching safely through a restricted opening on a truss structure in the laboratory.

In this experiment, the end-effecter is moved in teleoperation mode by the operator using the joystick. The rectangular truss opening has dimensions 63.5 by 87.6 cm and depth of 53.3 cm. The operator attempts to enter the end-effector into the opening while hitting the vertical side of the opening which is aligned with the end-effector frame y-axis. The sensor on the $+x$ surface of the end-effecter detects the truss beam and "pushes" the end-effecter away to avoid collision. A similar behavior is observed when the end-effecter is commanded to hit the horizontal beam of the truss opening which is aligned with the x-axis of the end-effecter frame. This reflexive motion of the end-effecter is depicted in the composite picture shown in Figure 11. When the stand-off distances for the $\pm y$ sensors are increased beyond half of the opening width, it is observed that the end-effecter exhibits sustained oscillations. This is expected since it is now physically impossible to find an end-effector position such that both of the virtual forces in the $\pm y$ directions are zero.

Finally, a white backplane is installed inside the truss opening, The range sensor mounted inside the end-effecter housing detects this surface which is perpendicular to the z-axis of the end-effecter frame. This sensory data is then used to prevent collision with the surface by maintaining a stand-off distance between the end-effecter and the surface. This capability prevents the operator from erroneously commanding the end-effecter to penetrate inside the opening more than the predefine distance.

5 Conclusions

A new formulation and solution to the sensor-based collision avoidance problem is presented in this paper. Virtual forces are generated based on proximity of the end-effecter to worksite objects. These forces are then nullified by perturbing the commanded motion of the end-effecter. This approach transforms the collision avoidance problem to a force control problem, which is a well-understood classical problem in robotics.

It is important to note that the real-time software developed previously for contact control[7] was used *without any modifications* to perform the sensor-based collision avoidance experiments reported in Section 4. The force/torque sensor data representing the *actual* contact force was simply replaced by the *virtual* force generated in the software based on the proximity sensor data.

Sensor-based collision avoidance is an important step towards autonomous control of manipulators. It also protects the manipulator from erroneous commands issued by the operator, as well as prevents the arm from collision with unexpected obstacles. Current research is focused on extending the proposed approach to whole-arm collision detection and avoidance. Furthermore, integration of model-based and sensor-based data for on-line collision avoidance is presently under investigation.

6 Acknowledgment

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

7 References

[]] C. L. Boddy and J.D.Taylor: "Whole-arm reactive collision avoidance control of kinematically redundant manipulators", Proc. IEEE Intern. Conf. on Robotics and Automation, pp. 382-387, Atlanta, May 1993.

[2] R. Volpe and J. Balaram: "Technology for robotic surface inspection in space", Proc. AIAA Conf. on Intelligent Robots in Field, Factory, Science, and Space, Houston, March 1994.

[3] E. Cheung and V. J. Lumelsky: "Proximity sensing in robot manipulator motion planning: system and implementation issues", IEEE Trans. on Robotics and Automation, pp. 740-751, 1989.

[4] J. T. Feddema and J. L. Novak: "Whole arm obstacle avoidance for teleoperated robots", Proc. IEEE Intern. Conf. on Robotics and Automation, pp. 3303-3309, San Diego, May 1994,

[5] T. S. Wikman and W. S. Newman: "A fast, on-line collision avoidance method for a kinematically redundant manipulator based on reflex control", Case Western Reserve University,

CAISR Technical Report 91-160, September 1991.
[6] J. J. Craig: "Robotics - Mechanics and Control", Addison-Wesley Publishing Company, 1989.
[7] 11. Seraji, D.Lim, and R. Steele: "Experiments in contact control", JPL Intern. Publ., 1994.

Document 7"

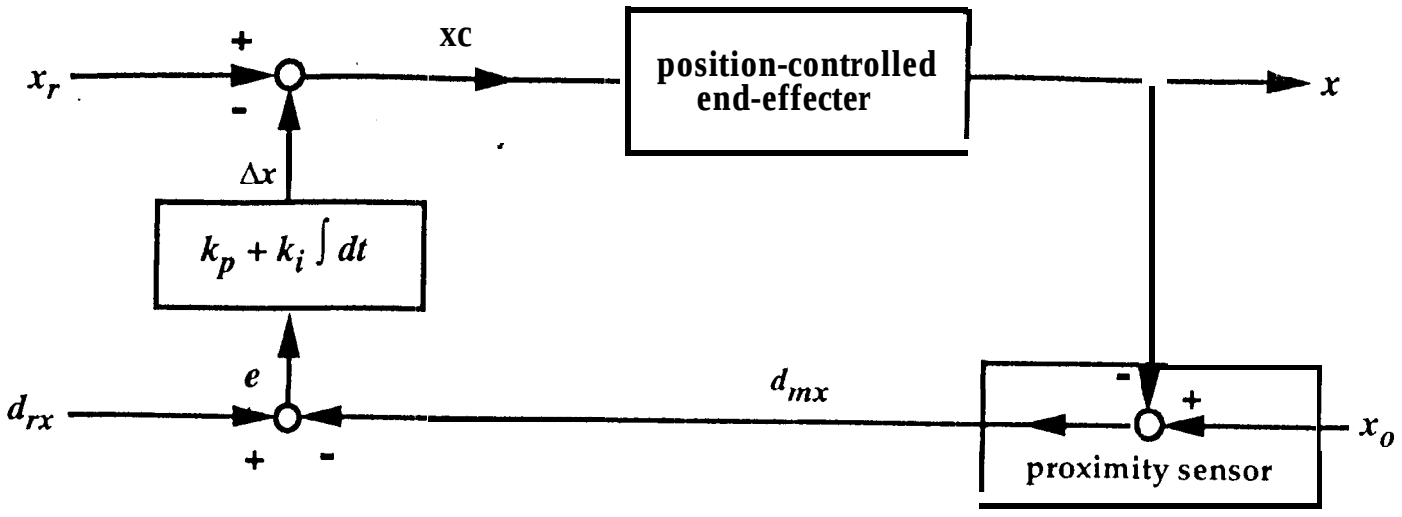


Figure 1: Sensor-Based Ekd-Effecter Collision Avoidance System for $d_{mx} \leq d_{rx}$

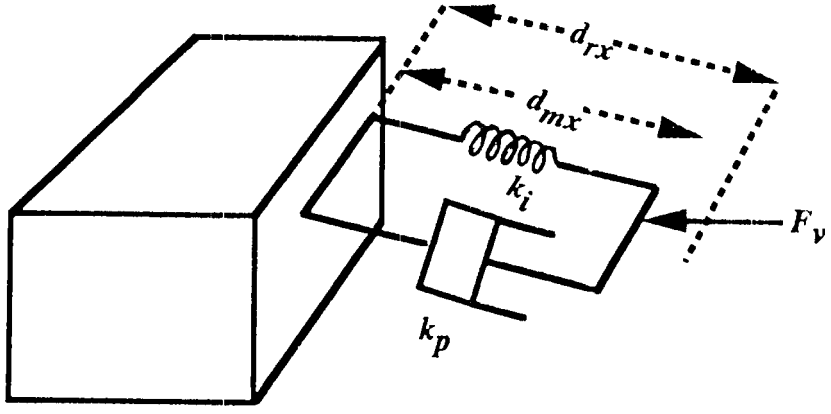


Figure 2a: Hypothetical spring-plus-damper attached to the object

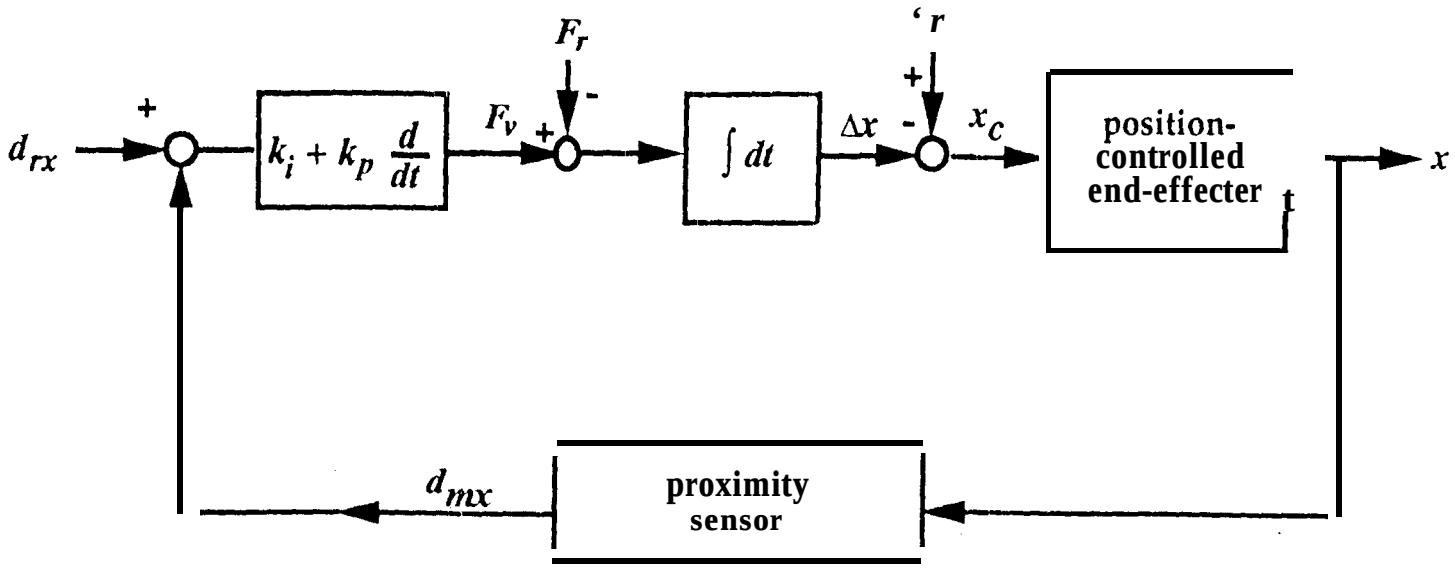


Figure 2b: Virtual Force Control Interpretation of Collision Avoidance

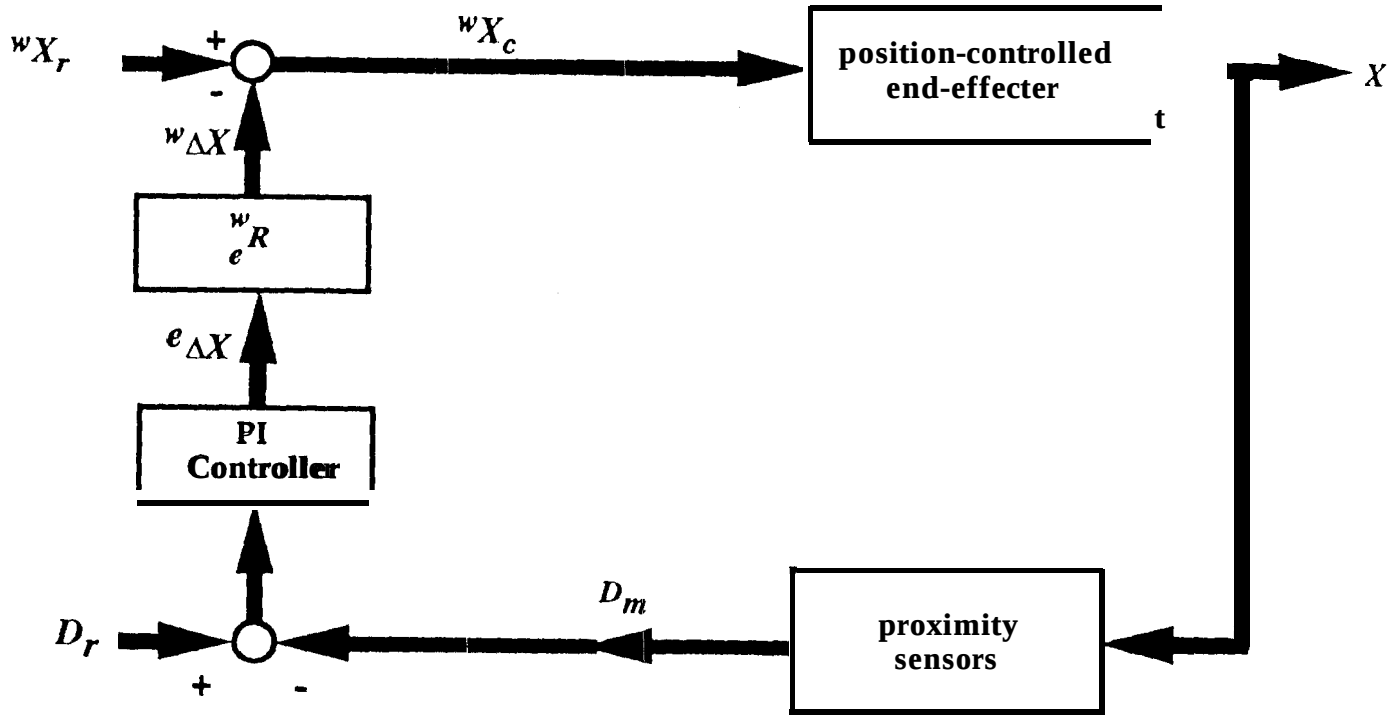
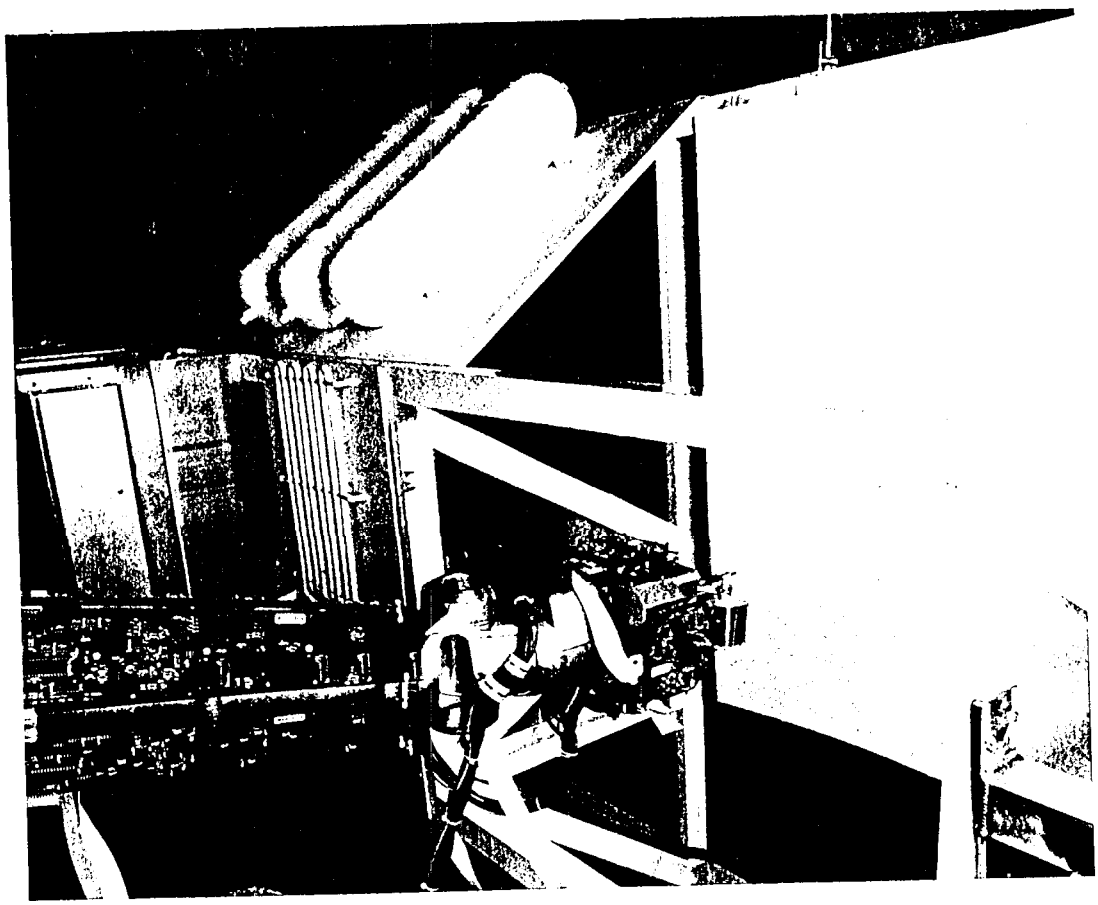
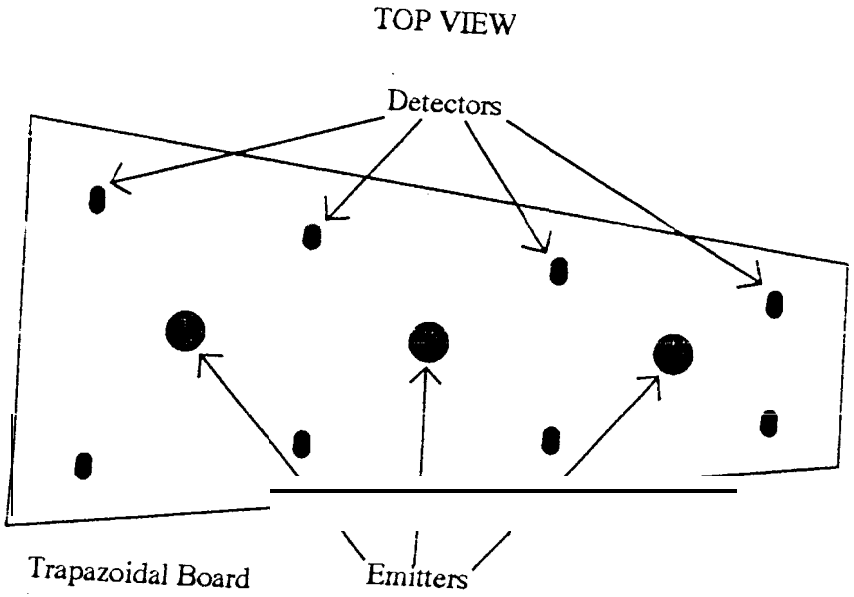


Figure 3: Implementation Diagram of Collision Avoidance System

Figure 4: Mounting of Sensor Skin cm RRC An-n



Figure❑ Layout of Each Sensor Cell



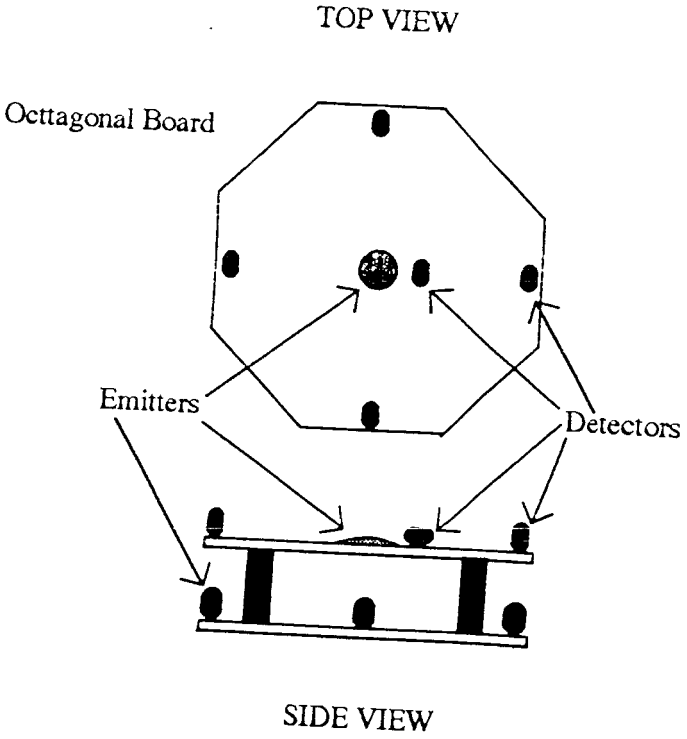


Figure 6: Layout of the Sensor Skin Boards

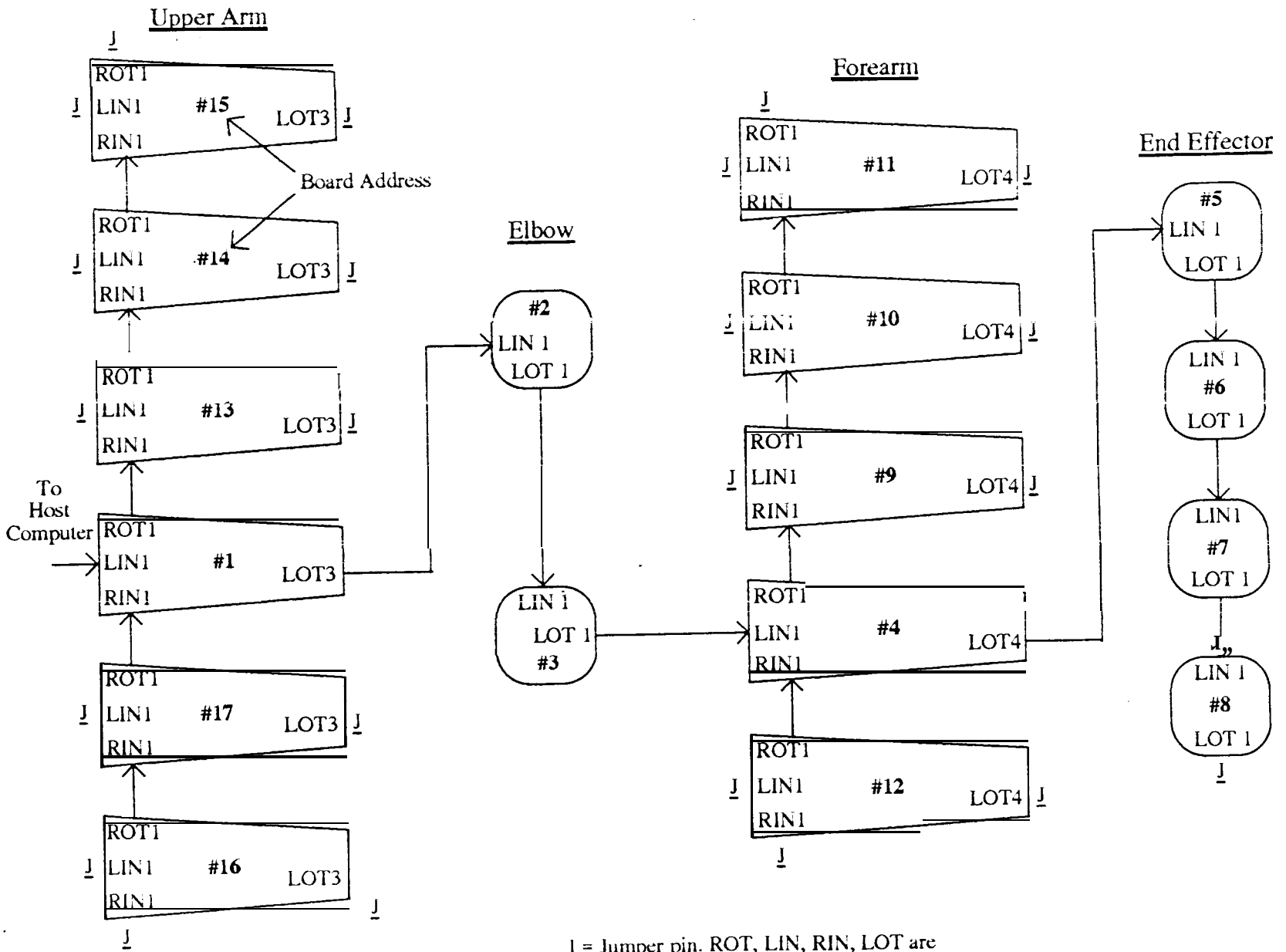


Figure 7: Calibration Plot of the Sensor Board 7

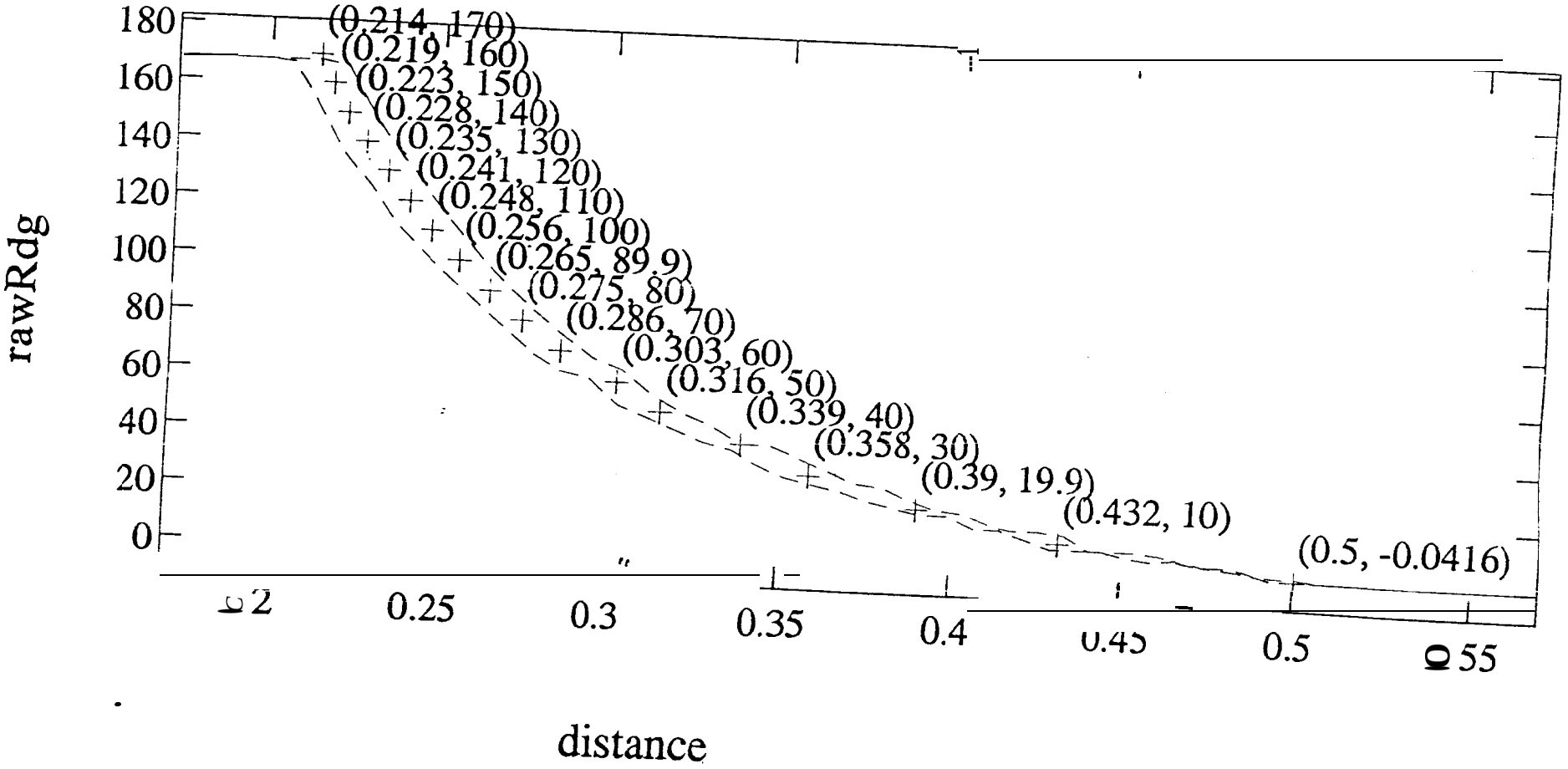


Figure 8: Error Plot of the Sensor Board 7

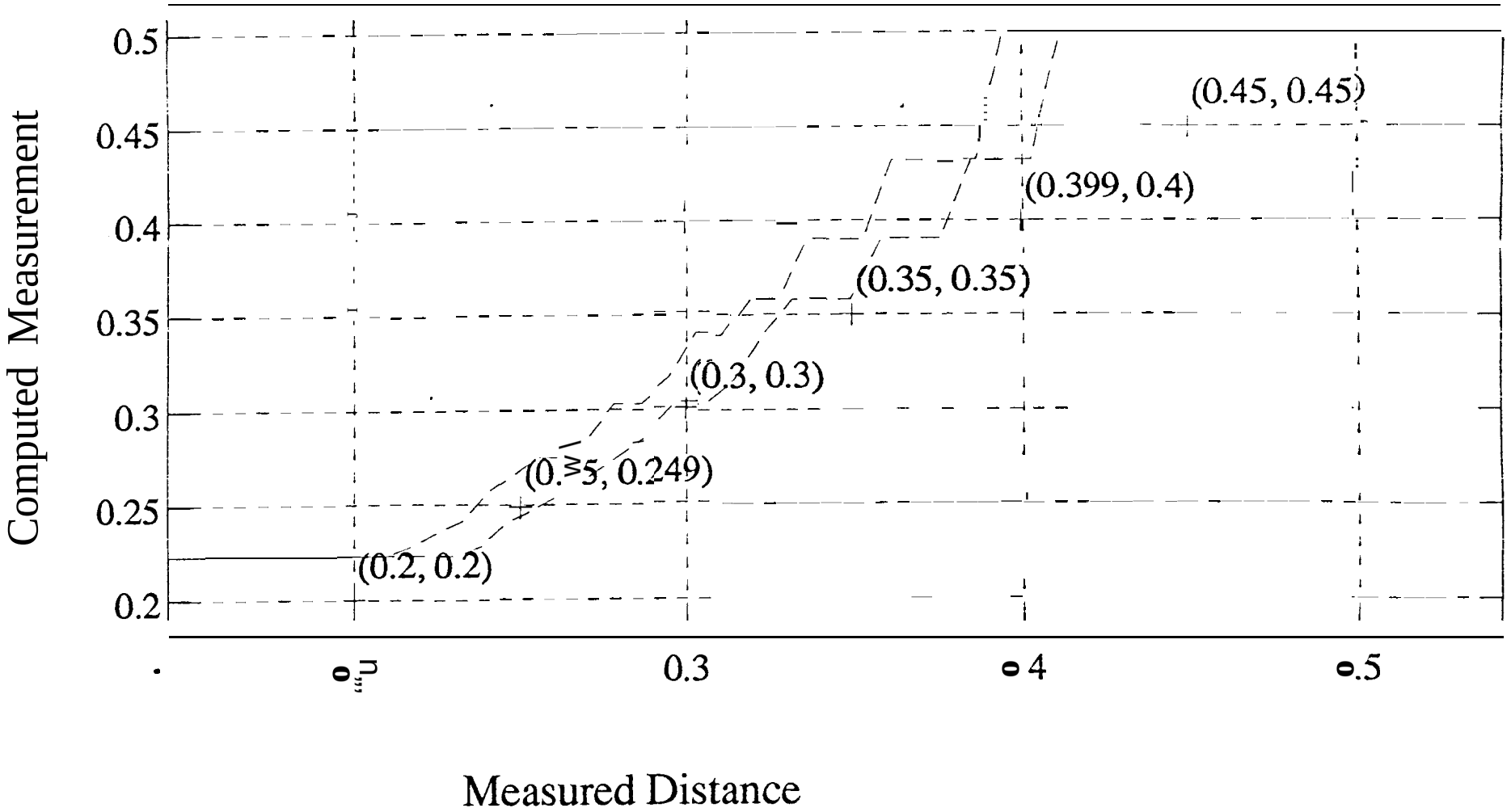


Figure 9a: End-Effector Displacement in x Direction for Collision Avoidance

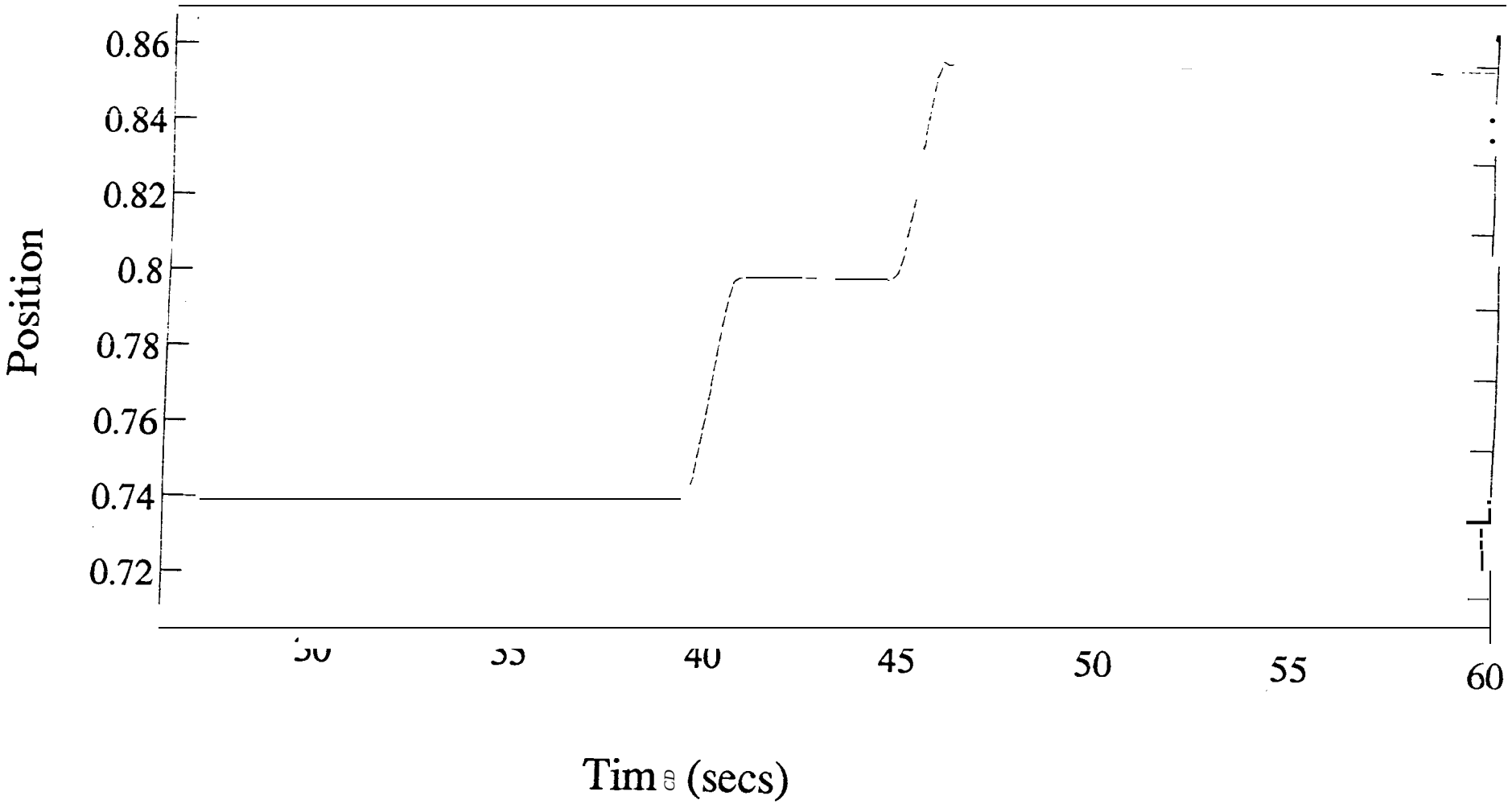


Figure 9b: Variation of Virtual Force with Object Distance

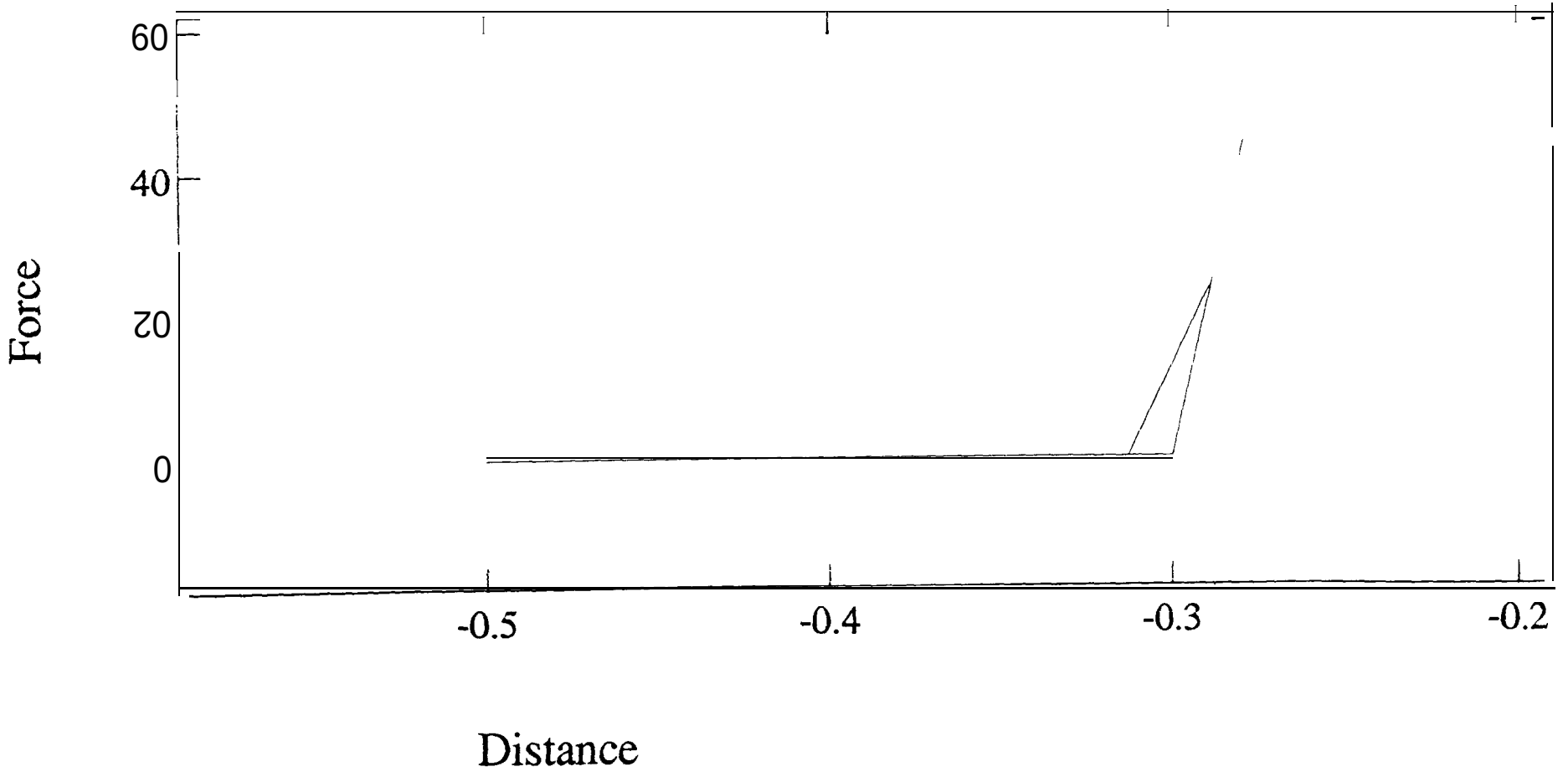


Figure 9c: Vari tion of Virtual Force with Time

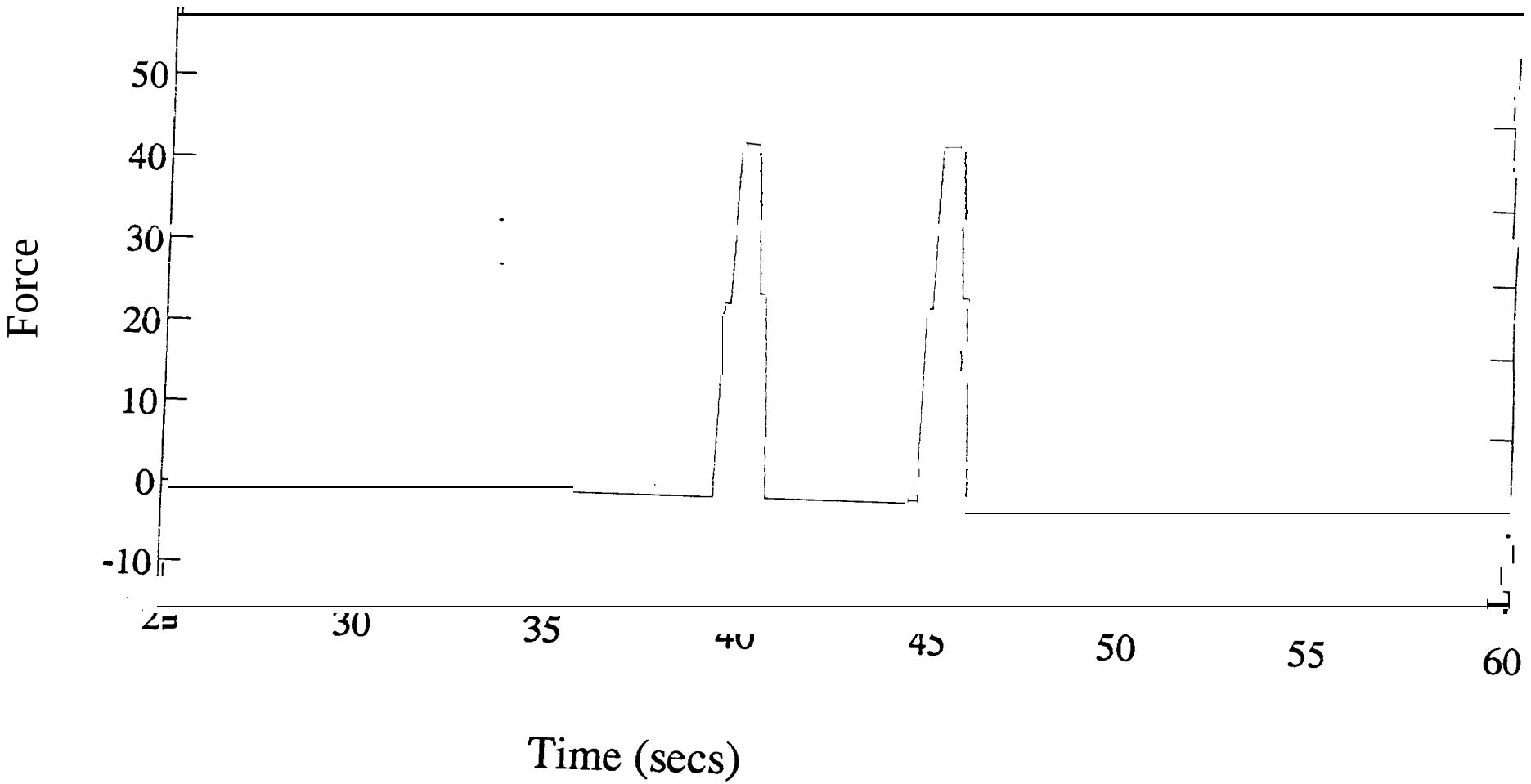


Figure 10a: End-Effector Displacement in *y* Direction for Collision Avoidance

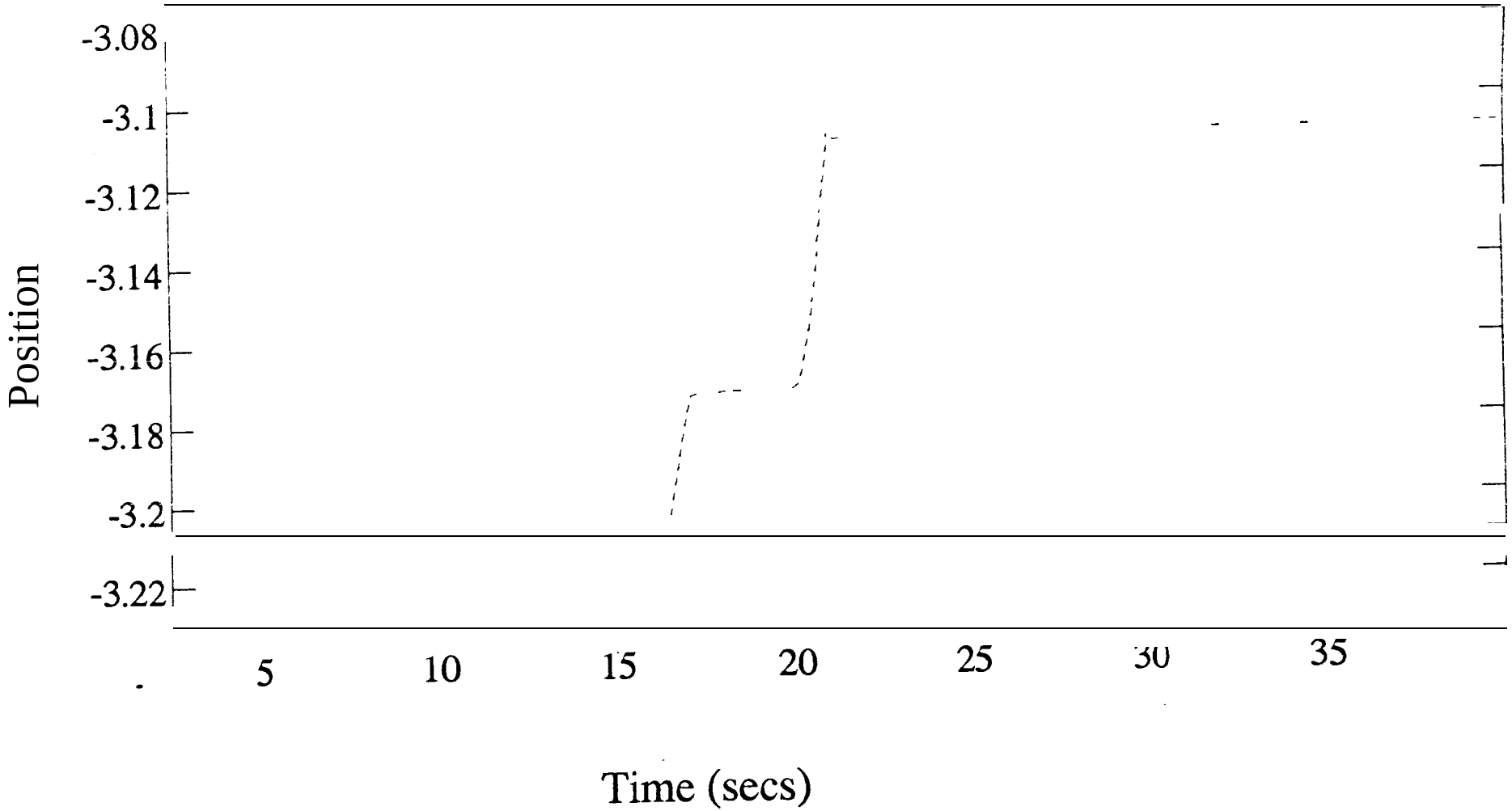


Figure 10b: End-Effector Displacement in z Direction for Collision Avoidance

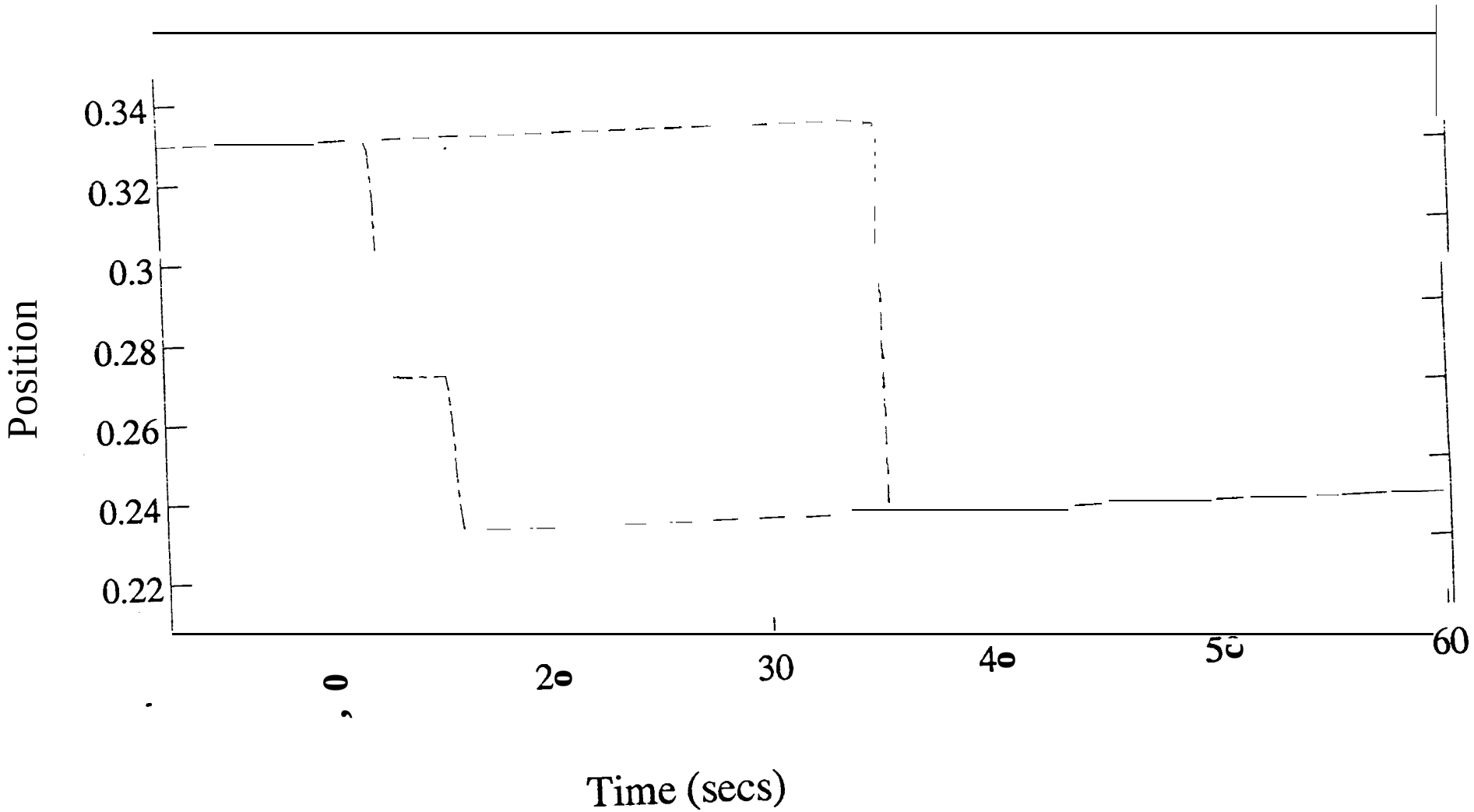


Figure 11: Composite Picture of lhd-Effecter Motion for Collision Avoidance